Numismatic Application>

Supplementary Specification

Version 1.0

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Description** | **Author** |
| 26/Mar/18 | 1.0 | <details> | Urda Sebastian George |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Table of Contents

1. Introduction 4

2. Non-functional Requirements 4

2.1 Availability 4

2.2 Performance 4

2.3 Security 4

2.4 Testability 4

2.5 Usability 4

3. Design Constraints 4

Supplementary Specification

# Introduction

The purpose of this document is to define requirements of the Numismatic Application system. This Supplementary Specification lists the requirements that are not readily captured in the use cases of the use-case model. The Supplementary Specifications and the use-case model together capture a complete set of requirements on the system.

This Supplementary Specification applies to the Numismatic application system which will be developed by me as my project for the end of the semester.

This specification defines the non-functional requirements of the system.

# Non-functional Requirements

## Availability

The Numismatic application can be available on any PC having windows 8 or higher as operating system with an internet connection. The lack of internet or a poor connection and some not very likely updates reduce the availability of the app to 90%

## Performance

The most difficult part in making the system performant will be to make the connection and the work with the database fast enough. At most a wait time of magnitude of 10s of seconds is acceptable. Expected to

have a latency induced when multiple users will be logged in into the web application.

## Security

The most important part will be to store the passwords and usernames safely in a database.

The application should have defense mechanisms for sql injection.

## Testability

The interaction with the database will be most important part of the testing. The project is simple enough that it can be tested easily. Scenarios to test are quite a lot but should be easy to implement. The application can have multiple testing layers, such as unit tests, integration tests, functional testing and so on.

## Usability

A friendly GUI will be created in order to make the application easy to use and intuitive.

# Design Constraints

For this project I will use an SQL database and Eclipse Java Neon and a client server pattern.